Improved Shape-Signature and Matching Methods for Model-Based Robotic Vision

J.T. Schwartz and H.J. Wolfson
New York University
New York, NY 10012

1. Abstract

We describe new techniques for curve matching and model-based object recognition, which are based on the notion of shape-signature. The signature which we use is an approximation of pointwise curvature. The talk will describe a curve matching algorithm which generalizes a previous algorithm due to Schwartz and Sharir which was developed using this signature allowing improvement and generalization of a previous model-based object recognition scheme. The results and the experiments to be described relate to 2-D images. However natural extensions to the 3-D case exist and are being developed.

2. Introduction.

The purpose of this talk is to survey the work done in the Robotics Laboratory of the Courant Institute on *Model-Based Object Recognition* with emphasis on recent results. Recognition of industrial parts and their location in a factory environment is a major task in robot vision. Most industrial part recognition systems are model-based systems (see a survey in [1]). The model based approach is well suited for an industrial environment, since the objects processed by the robot are usually known in advance, and belong to a certain subset of the factory's tools and products.

We discuss the 2-D object recognition problem, where the robot is faced with a composite scene of overlapping parts (thus partially occluding each other), taken from a large data-base of known objects (e.g. the factory's warehouse). The task is to recognize the objects in the scene and their location. We want the recognition time to be fast and depend on the size of the scene, which is usually small, and not on the size of the initial large data-base.

The algorithms which we describe were actually tested in a "real life situation" by recognition of objects comprising composite scenes of about ten thin overlapping cardboard pieces taken from a data-base of hundred pieces. In our approach the camera is held at a constant height over the scene, i.e is suitable for a conveyor belt situation.

Since we are concerned with recognition of overlapping objects, we cannot make use of global features such as area, perimeter or centroid of a 2-D object. However, since a 2-D object is fully described by its boundary curve, both globally and locally, we can use these curves in our recognition process. This requires development of robust and efficient curve matching algorithms. These algorithms are applicable not only for object recognition tasks, but also to other tasks where curve matching is required.

3. Preprocessing and Data Acquisition

We begin with three major preprocessing steps :

1) Planar pieces are photographed by a black and white RCA 2000 camera, and the pictures are digitized and thresholded to get a binary image for each piece.
2) The boundary of each piece is extracted from the binary image. These boundary curves are our "experimental" curves.
3) A *smoothing* procedure is applied to each curve. We use the procedure which is described in detail in [2]. Basically, this expands the noisy curve to a narrow strip defined by a certain threshold value ϵ and then finds the shortest path lying in this 2ϵ-wide strip.(It may be imagined as stretching a loose rubber band within a narrow sleeve.) This gives a polygonal approximation of each observed curve.

4. The Schwartz-Sharir Curve Matching Algorithm

The first algorithm we are going to describe is due to Schwartz and Sharir (see [2]). Given a curve and a proper subcurve of it in the plane, it computes the rotation and translation of the subcurve relative to the curve which gives the best match in an L_2 kind of metric. Moreover it also computes the distance between the two aligned curves in this metric, thus giving us a score of the quality of the proposed match.

Take two curves C and C' in the plane and assume that C is a translated and rotated subcurve of C'. Both C and C' are assumed to have been smoothed (i.e. they are polygonal approximations of the original curves) and parametrized by arc length s. The matching we seek calls for determination of the offset s_0 and the Euclidean transformation E for which the curves $EC(s)$ and $C'(s+s_0)$ are closest to one another in the L^2 norm. Specifically, we represent each of the curves C, C' by a sequence of evenly spaced points on it, and let these sequences be $(\mathbf{u}_j)_{j=1}^n$ and $(\mathbf{v}_j)_{j=1}^m$ respectively. Assume first that both curves have the same starting point (i.e. $s_0=0$ and, hence, $m \geq n$). Matching amounts to finding a Euclidean motion E of the plane which will minimize the l^2 distance between the sequences $(E\mathbf{u}_j)_{j=1}^n$ and $(\mathbf{v}_j)_{j=1}^n$:

$$\Delta = \min_E \sum_{j=1}^{n} |E\mathbf{u}_j - \mathbf{v}_j|^2$$

To simplify the calculation, first translate C so that

$$\sum_{j=1}^{n} \mathbf{u}_j = 0$$

Next write E as $E\mathbf{u} = R_\theta \mathbf{u} + \mathbf{a}$, R_θ denoting a counterclockwise rotation by θ.
In such case, as it is shown in [S-S], the best match is obtained when

$$\mathbf{a} = \frac{1}{n} \sum_{j=1}^{n} \mathbf{v}_j$$

and θ is the negation of the polar angle of $\sum u_j \bar{v}_j$, where the vectors $\mathbf{u}_j$, $\mathbf{v}_j$ are regarded as complex numbers u_j, v_j. The least-square distance for this best match is given by

$$\Delta = \sum_{j=1}^{n} |\mathbf{v}_j|^2 - \frac{1}{n} \left| \sum_{j=1}^{n} \mathbf{v}_j \right|^2 + \sum_{j=1}^{n} |\mathbf{u}_j|^2 - 2 \left| \sum_{j=1}^{n} u_j \bar{v}_j \right| \qquad (*)$$

If the curves do not have the same starting point, we have to match the sequence $(\mathbf{u}_j)_{j=1}^n$ to each of the contiguous subsequences $(\mathbf{v}_{j+d})_{j=1}^n$ of the sequence $(\mathbf{v}_j)_{j=1}^m$, for $d = 0, \ldots, m-n$.
For each such d (*) thus becomes

$$\Delta(d) = \sum_{j=d+1}^{d+n} |\mathbf{v}_j|^2 - \frac{1}{n} \left| \sum_{j=d+1}^{d+n} \mathbf{v}_j \right|^2 + \sum_{j=1}^{n} |\mathbf{u}_j|^2 - 2 \left| \sum_{j=1}^{n} u_j \bar{v}_{j+d} \right|$$

We seek the minimum of the values $\Delta(d)$, $d = 0, \ldots, m-n$, which can be found in time $O(m \log m)$, using the fast Fourier transform algorithm for computing the convolutions $\sum_{j=1}^{n} u_j \bar{v}_{j+d}$.

The algorithm described above is a corner-stone of our subsequent techniques. It has two major advantages - speed and robustness, as demonstrated in an experiment of a successful computerized assembly of 100 piece jigsaw puzzles with a lot of almost similar pieces (see [3]). (The algorithm was used as the local curve matching algorithm between the boundary curves of the pieces providing scores to the goodness of this matches, and then a global algorithm, based on combinatorial optimization techniques, used this scores to compute the correct solution of the puzzle.)

The curve matching algorithm can be applied directly to give a first solution of the model-based object recognition problem. First we have to divide the boundary of our composite scene into subcurves, such that each such subcurve is supposedly part of the boundary curve of a different object in the scene. This can be done most simply by assuming that objects in a scene meet at sharp concave angles (the so called "breakpoints" in [4]). Then each such subcurve can be matched against the boundary curves of all the objects in our data-base to determine the best matching object. However, this approach has two serious drawbacks in regard to our original goals. First, the use of "breakpoints" may in some cases cause a wrong subdivision of the boundary of the composite scene; secondly, the speed of recognition grows linearly with the size of the data-base, indicating the need for a more efficient technique. Thus further development is required, and the following sections will describe solutions to these problems. However, as was mentioned before, this algorithm is an essential part of the later developed methods, mainly because of its robustness.

5. The Generalized Curve Matching Algorithm

In order to avoid the use of the "breakpoint" heuristic, and also be able to solve a number of other curve matching problems we require an efficient solution to the following more general curve matching problem:

given two curves, find the longest matching subcurve which appears in both curves.

An approach to this problem due to Wolfson (see [6]) can be summarized as follows :

Step A : Represent both curves by *characteristic strings* which represent local translationally and rotationally invariant features.

Step B : Find the longest common substring of the two characteristic strings, and also find other long common substrings if these are nearly as long.

Step C : For each substring produced by *Step B* , go back to the original curves and match the two subcurves which correspond to this substring using the preceding subcurve matching algorithm, thus determining the desired translation and rotation of one curve with respect to the other.

Step D : Rotate and translate the curves accordingly, and determine again the longest matching subcurves of the two curves, given this rotation and translation. This subcurve is found by simply checking the (x,y) coordinates of corresponding points on the curves and demanding that the distance between the points should be less than a certain threshold value ϵ. This final check works with points on the curves themselves, and not with the (less accurate) feature string values at these points; hence it is quite robust.

The result giving the longest matching subcurve (allowing minor mismatches) is chosen as the final solution.

Two algorithms using this approach were developed, reflecting a certain trade-off between robustness and the theoretical complexity of these proposed techniques. One of them uses efficient string matching techniques due to Weiner and to McCreight (see [7],[8]) to find the long matching substrings in time which is linear in the length of the strings. This makes the whole algorithm linear in the number of sample points on both curves, since the information obtained at *Step B* allows the

curve matching algorithm to be implemented in linear time as well. However, the string matching algorithms mentioned above require the string elements with which they work to be taken from a finite alphabet, which forces us to truncate the feature strings (which consist of real numbers). This may cause otherwise long matching substrings to split into a number of shorter matching strings. To overcome this problem we developed a string matching algorithm which regards string elements equal when the difference between the two elements is less than a certain threshold value ϵ. This algorithm can be implemented in time which is proportional to $Max(n \log n, \epsilon n^2)$, thus making this algorithm quite efficient for curves of practical length. Experiments have shown this algorithm to be both efficient and robust.

This approach enables us to solve the object recognition problem by matching the boundary curve of the composite scene against the boundary curves of the objects in the data-base. Objects having long subcurves matching the composite scene and satisfying obvious consistency requirements will be objects participating in the scene. However this approach still does not satisfy the efficiency goal that we have set, since it is linearly dependent on the number of objects in the data-base. Thus additional ideas must be applied to the solution of the object recognition problem.

6. Shape Signatures

The method described in the previous section uses local rotationally and translationally invariant features to characterize boundary curves. In this section we will examine one such feature.

Our aim is to represent any curve C by a characteristic string of reals $(c_i)_{i=1}^n$. Since these strings will be compared to achieve subcurve matching, we want the numbers $(c_i)_{i=1}^n$ to encode characteristics of the curve which are :

i) local,

ii) translationally and rotationally invariant,

iii) stable, in the sense that small changes in the curve induce small effects (or no effect at all) in the associated sequence $(c_i)_{i=1}^n$,

a further desirable, but less essential, property is :

iv) an approximation to an observed curve can be reconstructed from its characteristic string.

One "natural" feature which satisfies these conditions is the pointwise *curvature* of a curve (see Chapter II of [9]). It is well known that there is a one to one correspondence between a regular curve (modulo translation and rotation) and its curvature function (which is a continuous function of its arclength). However, our applications must deal with noisy polygonal representations of curves, making it impossible to compute curvatures either accurately, or at every point of a curve. Thus we must work with an approximation of the curvature, calculated at discrete points of the curve, to get a data sequence $(c_i)_{i=1}^n$ having the desired properties.

Let $\kappa(s)$ be the curvature function of a curve C, where s denotes arclength along the curve. $\kappa(s)$ is the derivative of the tangent angle $\theta(s)$ to the curve, parametrized as a function of its arclength. To approximate the curvature, we first build the so called *arclength versus turning angle* graph of the curve C. (Since after our smoothing procedure we have a polygonal approximation of the observed curve, this is a step function.) Then we sample this graph at equally spaced points, and at every such point s_i ($i = 1,\ldots,n$) we compute the difference

$$\Delta\theta(s_i) = \theta(s_i + \Delta s) - \theta(s_i)$$

(To make the method more robust we actually compute an averaged difference

$$\Phi_i = Av\Delta\theta(s_i) = \frac{1}{k}\sum_{j=0}^{k-1} \Delta\theta(s_i + j\delta)$$

Detailed choice of the parameters Δs, k, δ is based on experimental considerations.)

Remark: The averaged differences (ϕ_i) satisfy the conditions (i)-(iii) required of local curve characteristics.

At first glance algorithms based on such features would not seem to be robust, since we are computing approximations of a second derivative of an initially noisy function. Thus this kind of signature needs to be applied with care, mainly in preliminary steps which aim simply to filter out obvious "wrong" candidates in an efficient way, to prepare for final decisions made using more robust procedures. Note that in the previous algorithms described the use of the feature strings was quite limited: we used it to locate approximate starting points and endpoints of several long candidate subcurves for matching. Matching itself is done using the robust subcurve matching algorithm.

In the following section we will describe another use of shape signatures which solves the object recognition problem in a still more efficient way.

7. The "Footprint" Approach

In this section we give a method which solves the object recognition problem in a particularly efficient way. The method was first developed by Kalvin, Schonberg, Schwartz and Sharir in [4] and later improved by Hong and Wolfson (see [5]). We will present the later version, which differs from the previous one in two aspects - it does not require use of "breakpoints" in advance to divide the boundary curve of the composite scene into subcurves belonging to different objects, and it uses shape signatures based on approximate local curvatures, rather than the Fourier descriptors used in the previous version.

The algorithm consists of two major steps. The first one is a preprocessing step which is done on the data-base of model objects to be recognized. The complexity of this step is linear in the size of the data-base. This step can be executed off-line before actual recognition is needed. The second step, recognition proper, uses the data prepared by the first step and can be executed in time which on the average is linearly dependent on the size of the composite scene, thus achieving recognition time almost independent on the size and number of objects in the data-base.

A) Preprocessing

All the objects in the data-base are processed as follows. The boundary curve of every object is scanned and shape signature values are generated at equally spaced points. (We use a 5-tuple of approximate local curvatures at consecutive points.) This "footprint" is local, translationally and rotationally invariant. For each such footprint we record the object number and the sample point number at which this footprint was generated. (This data is held as a hash-table.) The "footprint" data points on the boundary of the object have a natural order, which is defined by the way the boundary curve is traced. This preprocessing step is linearly dependent on the total of sample points on the boundary curves of all the objects in the data-base. New objects added to the data-base can be processed independently without recomputing the hash-table (except when we must change its size by re-hashing).

B) Recognition

In the recognition stage the boundary curve of the composite scene is scanned and footprints are computed at equally spaced points. For each such footprint we check the appropriate entry in the hash-table, and for every pair of object number and sample point number, which appears there, we tally a vote for the object and the relative shift between the object and the scene. For example, if a footprint, which was computed at the i'th sample point on the composite scene, appeared on objects k_1 and k_2 at sample points j_1 and j_2 respectively, we add votes to object k_1 with relative shift $i-j_1$ and object k_2 with relative shift $i-j_2$.

At the end of this process we find those (object,shift) pairs that got most of the votes, and for every such pair determine approximate starting and endpoints of match between the footprint string of the composite scene and the footprint string of the object under the appropriate shift. Given these matching substrings we may

apply the robust subcurve matching algorithm described previously. This process resembles that used in the generalized curve matching algorithm presented in section 5. Once the object which has the longest matching subcurve with the composite scene is discovered, we decide that it is one of the objects in the scene, discard its matching subcurve, and repeat the process for the remaining curves in the reduced composite scene. At this stage a number of objects can be processed simultaneously.

The algorithm described is on the average linear in the number of sample points in the composite scene, and does not depend directly on the number of points on the boundaries of all the objects in the data-base. Thus we achieve the efficiency goal set at the beginning.

An improvement of this method can be achieved by the introduction of "weighted" footprints. Since for typical curves in different environments not all the footprints have an equal probability of occurrence, it seems desirable not to give an equal weight to every "hit", but to give a higher weight to coincidence of "rare" footprints. The actual probability of an individual footprint can be estimated by the number of its occurrences in the data-base, which can serve as a statistical sample for this kind of data. The weighted footprint approach can also improve its efficiency by assigning zero weight to very frequent footprints and thus saving us the need to process hash-table entries with a lot of candidates. These entries require much computer time but contribute only a small amount of information.

The method described above has proved to be quite robust. It also generalizes previously used methods based on use of special boundary features such as sharp angles. In our approach this is a special case, these special features being assigned large weight, while other features get zero weight. An appropriately sophisticated weight function can benefit from all the available information, and can deal with scenes which have no sharp angles or any other distinctive features, which are known in advance.

A major potential advantage of our "footprint" algorithm is its high inherent parallelism. Parallel implementation of this algorithm is straightforward; moreover, it should be quite easy to build a special device for this implementing it at very high speed.

8. 3-D Curve Matching

All the algorithms described here apply just as well to 3-D curve matching. The subcurve matching algorithm of section 3 has been implemented in the 3-D case (see [10]) and seen to perform well; currently a generalized curve matching algorithm, which uses 3-D shape signatures is being implemented.

9. Acknowledgements

Work on this paper was supported by Office of Naval Research Grant N00014-82-K-0381 and National Science Foundation Grant No. NSF-DCR-83-20085.

REFERENCES

[1] R.T. Chin and C.R. Dyer, *Model-Based Recognition in Robot Vision*, Computing Surveys, Vol. 18, No.1, March 1986, pp.67-108.

[2] J.T. Schwartz, and M. Sharir, *Identification of Partially Obscured Objects in Two Dimensions by Matching of Noisy "Characteristic Curves"*, Tech. Rep. No.165, June 1985, Comp. Sc. Div., Courant Inst. of Math., NYU (to appear in the Int. J. of Robotics Research).

[3] A. Kalvin, Y. Lamdan,E. Schonberg, and H. Wolfson, *Solving Jigsaw Puzzles Using Computer Vision*, Tech. Rep. No.205, Feb. 1986, Comp. Sc. Div., Courant Inst. of Math., NYU (to appear in "Approaches to Intelligent Decision Support" in the series Annals of Operations Research (1987))

[4] A. Kalvin, E. Schonberg, J.T. Schwartz, and M. Sharir, *Two Dimensional Model Based Boundary Matching Using Footprints*, Tech. Rep. No.162, May 1985, Comp. Sc. Div., Courant Inst. of Math., NYU (to appear in the Int. J. of

Robotics Research 5(4) (1986)).

[5] J.W. Hong and H. Wolfson, *On Footprints,* in preparation.

[6] H. Wolfson, *On Curve Matching,* Tech. Rep. No.256, Nov. 1986, Comp. Sc. Div., Courant Inst. of Math., NYU.

[7] P. Weiner, *Linear Pattern Matching Algorithms* , 14'th Annual Sym. on Switching & Automata Th.(Iowa), IEEE Comp. Soc.,(1973), pp. 1-11.

[8] M. McCreight, *A Space-Economical Suffix Tree Construction Algorithm,* J. of the ACM, Vol.23, No.2, April 1976, pp. 262-272.

[9] J.J. Stoker, *Differential Geometry,* Wiley-Interscience, 1969.

[10] M. Bastuscheck, E. Schonberg, J.T. Schwartz, and M. Sharir, *Object Recognition by 3-Dimensional Curve Matching,* Tech. Rep. No.203, Jan. 1986, Comp. Sc. Div., Courant Inst. of Math., NYU (Int. J. of Intelligent Systems (1986)).